Daniel Gottesman *and* Isaac Chuang

Computer Science Division, University of California, Berkeley, CA 94270
MIT Media Laboratory, Cambridge, MA 02139

The physics of quantum systems opens a door to tremendously intriguing possibilities for cryptography, the art and science of communicating in the presence of adversaries [1]. One major goal of classical cryptography is to certify the origin of a message. Much like a handwritten signature on a paper document, a *digital signature* authenticates an electronic document and ensures that it has not been tampered with. The importance of digital signatures to modern electronic commerce has become such that Rivest has written "[they] may prove to be one of the most fundamental and useful inventions of modern cryptography." [2] This is especially true of schemes where the signature can be recognized using a widely available reference. The security of all such public key digital signature schemes presently depends on the inability of a forger to solve certain difficult mathematical problems, such as factoring large numbers [3]. Unfortunately, with a quantum computer factoring becomes tractable [4], thus allowing signatures to be forged. Here, we present a *quantum* digital signature scheme which is absolutely secure, even against powerful quantum cheating strategies. It allows a sender (Alice) to sign a message so that the signature can be validated by one or more different people, and all will agree either that the message came from Alice or that it has been tampered with.

Classical digital signature schemes can be created out of any one-way function [5]. $f(x)$ is a one-way function if it is easy to compute $f(x)$ given x, but computing x given $f(x)$ is very difficult. This allows the following digital signature scheme [6]: Alice chooses k_0 and k_1, and publicly announces f, $(0, f(k_0))$ and $(1, f(k_1))$. Later, to sign a single bit b, Alice presents (b, k_b). The recipient can easily compute $f(k_b)$ and check that it agrees with Alice's earlier announcement, and since k_0 and k_1 were known only to Alice, this certifies that she must have sent the message. However, while there are many candidate one-way functions, none have been proven to be secure, and some, such as multiplying together two primes (the inverse being factoring the product), become insecure on a quantum computer. This deficiency leaves a substantial gap in the cryptographic landscape.

Our quantum digital signature scheme is based on a quantum analogue of a one-way function which, unlike any classical function, is provably secure from an information-theoretic standpoint, no matter how advanced the enemy's computers. The key idea we introduce is a one-way function whose input is a classical bit-string k, and output is a *quantum* state $|f_k\rangle$ (versus, for instance, a function which maps quantum states to quantum states). Like the above classical scheme, we will require $O(m)$ qubits to sign a m-bit message. It is not sufficient, however, to simply plug in $|f_k\rangle$ in place of $f(k)$. First, due to the no-cloning theorem, there can be no perfect equality test for quantum states. Also, as we show below, the nature of quantum states provides Alice with non-classical cheating strategies, and eavesdroppers with non-classical forgery mechanisms. And unlike classical schemes, only a limited number of copies of the public key can be issued, or the scheme becomes insecure. Despite these difficulties, the protocol we present, when used correctly, allows the probability of any security failure to be made exponentially small with only polynomial expenditure of resources.

Let us begin with the quantum one-way function. Suppose we take all classical bit strings k of length L, and assign to each one a quantum state $|f_k\rangle$ of n qubits. Further, let the states be nearly orthogonal: $|\langle f_k|f_{k'}\rangle| \leq \delta$ for $k \neq k'$; this allows L to be much larger than n. Buhrman, Cleve, Watrous, and de Wolf introduced one such family as *quantum fingerprints*, in which $L = O(2^n)$ and $\delta \approx 0.9$ [7]. Another family is provided by the set of stabilizer states [1], with $L = n^2/2 + o(n^2)$, and $\delta = 1/\sqrt{2}$. Both these sets are easy to create with any standard set of universal quantum gates. A third family of interest uses just $n = 1$ qubit per state, and consists of the states $\cos(j\theta)|0\rangle + \sin(j\theta)|1\rangle$, for $\theta = \pi/2^L$, and integer j. This family works for any value of L, and gives $\delta = \cos\theta$.

The mapping $k \mapsto |f_k\rangle$ acts as a sort of "quantum one-way function" because it is impossible to invert, but easy to compute and verify. Holevo's theorem puts limits on the amount of classical information that can be extracted from a quantum state [8]; in particular, measurements on n qubits can give at most n classical bits of information. Thus, given t copies of the state $|f_k\rangle$, we can learn at most tn bits of information about k, and when $L - tn \gg 1$, our chance of successfully guessing the string k remains small.

We take for granted certain properties of classical functions which are no longer so straightforward quantum-mechanically. Given two outputs $|f_k\rangle$ and $|f_{k'}\rangle$, how can we be sure that $k = k'$? This is done using a simple quantum circuit [7], which we shall call the *swap-test*. Take the states $|f_k\rangle$ and $|f_{k'}\rangle$, and prepare a single ancilla qubit in the state $(|0\rangle + |1\rangle)/\sqrt{2}$. Next, perform a Fredkin gate (controlled-swap) with the ancilla qubit as control and $|f_{k'}\rangle$ and $|f_k\rangle$ as targets. Then perform a Hadamard on the ancilla qubit and measure it. If the result is $|0\rangle$, then the swap-test is passed; this always happens if $|f_{k'}\rangle = |f_k\rangle$. Otherwise, if $|\langle f_{k'}|f_k\rangle| \leq \delta$, the result $|0\rangle$ occurs with probability at most $(1 + \delta^2)/2$. If the result is $|1\rangle$, then the test fails; this happens only when $k \neq k'$ and occurs with probability

$(1-\delta^2)/2$. Clearly the swap test works equally well even if the states are not outputs of the function f — if the states are the same, they always pass the swap test, while if they are different, they sometimes fail.

Another important property is the ability to verify the output of the function: given k, how do we check that a state $|\psi\rangle = |f_k\rangle$? This is straightforward: since the function $|k\rangle|0\rangle \mapsto |k\rangle|f_k\rangle$ is easy to compute, simply perform the inverse operation, and measure the second register. If $|\psi\rangle \neq |f_k\rangle$, the measurement result will be nonzero with probability $1 - |\langle\psi|f_k\rangle|^2$.

Blindly modifying classical cryptographic protocols to use quantum one-way functions will generally fail. First, given the output of a classical one-way function, someone with limited computational ability can learn nothing at all about the input, whereas $|f_k\rangle$ always leaks a limited amount of information about k, the input to the quantum one-way function. This is why in our signature scheme, Alice must limit the number of copies of her public keys in circulation. Second, verification of the identity of $|f_k\rangle$ can only be done with some error. Third, quantum cheating strategies become available; for example Alice (the person preparing the state) can prepare an entangled initial state, which enables her to delay choosing k until after she has given $|f_k\rangle$ away. This fact spells the doom of any attempt to use quantum one-way functions to perform bit commitment [9,10], which is one application of classical one-way functions. However, *only* Alice has the ability to change the state, which enables us to use quantum one-way functions to perform digital signatures.

Our digital signature protocol consists of two stages. The first step is the key distribution stage, where Alice creates and distributes quantum states which we shall refer to as her public keys. The public keys are "public," in the sense that no particular security measures are necessary in distributing them. If a number of copies fall into the hands of potential forgers, the protocol remains secure, provided the honest recipients receive valid keys. Classically, it is much easier to deal with identical public keys than with private keys that vary from recipient to recipient. The only purpose of our key distribution stage is to check that the public keys are truly indistinguishable. In the second stage of the protocol, Alice sends a classical message, and the t recipients use the public keys to verify that the message was sent by Alice. We shall initially describe how Alice sends one bit, b; multiple bits could be sent by repeating the protocol, but we describe a more efficient method at the end of the paper.

All participants in the protocol will know how to implement the map $k \mapsto |f_k\rangle$. All participants will also know two numbers, c_1 and c_2, thresholds for acceptance and rejection used in the protocol. A bound on the allowed value of c_2 will be given as part of the proof of security, below. c_1 can be zero in the absence of noise; the gap $c_2 - c_1$ limits Alice's chance of cheating. We assume perfect devices and channels throughout this paper, but our protocol still works in the presence of weak noise by letting c_1 be greater than zero, and with other minor adjustments.

The key distribution stage works as follows:

1. Alice creates a set $\{k_0^i, k_1^i\}$, $1 \leq i \leq M$, of pairs of L-bit strings. The k_0's will be used to sign 0's in the message, and the k_1's will be used to sign 1's. Note k_0^i and k_1^i are chosen independently and randomly for each i.
2. Alice creates $2t$ copies of each of the states $\{|f_{k_0^i}\rangle, |f_{k_1^i}\rangle\}$. These will be Alice's public keys.
3. Alice sends her public keys to a key distribution center, and each of the t recipients downloads two copies of each $\{|f_{k_0^i}\rangle, |f_{k_1^i}\rangle\}$. One copy will be used to verify the message, and one to test for Alice cheating. The public keys have been labelled by Alice, so the recipients know which key is which (but not the identities of the individual keys).
4. Finally, for each value of i, the recipients verify that they all received the same public keys using the swap test. Each recipient first performs a swap-test between their two keys, then each passes one copy to a single recipient. That recipient checks that these t test keys remain unchanged when any pair is swapped. If any of the public keys fail the test, the protocol is aborted. Otherwise, discard the test keys.

Assuming all recipients' public keys pass the swap test, ideally all the recipients would now have identical public keys. However, a dishonest Alice may create states which pass the swap test but are different for different recipients. Nevertheless, all the keys are equivalent in the sense that on average, each recipient will find a similar number of correct keys for a given message. Alice can now send a message b using the following procedure:

1. Alice sends the signed message $(b, k_b^1, k_b^2, \ldots, k_b^M)$ over an insecure classical channel. Thus, Alice reveals the identity of half of her public keys.
2. Each recipient of the signed message checks each of the revealed public keys to verify that $k_b^i \mapsto |f_{k_b^i}\rangle$. Recipient j counts the number of incorrect keys; let this be s_j.
3. Recipient j accepts the message if $s_j \leq c_1 M$, and rejects it if $s_j \geq c_2 M$. If $c_1 M < s_j < c_2 M$, the action taken by recipient j varies with the scenario. For instance, j might consult with the other recipients.
4. Discard all used and unused keys.

When s_j is large, the message has been heavily tampered with, and may be invalid. When it is small, the message cannot have been changed very much from what Alice sent. s_j is similar for all recipients, but need not be identical. As we shall see below, these tampering and forgery scenarios

are caught using the c_2 and c_1 thresholds. Forgery is prevented by c_2, and cheating by Alice is prevented by a gap between c_2 and c_1. Alice might attempt to divide the recipients, but she will almost always fail: she must mind the gap.

We prove the security of this scheme against two scenarios of cheaters. In the first scenario, only Alice is dishonest; her goal is to get recipients to disagree about whether a message is valid or not (i.e., she wishes to "repudiate" it). We show that if one recipient unconditionally accepts ($s_j < c_1 M$), then it is very unlikely that another will unconditionally reject ($s_{j'} > c_2 M$).

The second scenario is a standard forging scenario. In this case, Alice and at least one recipient Bob are honest. Other recipients or some third party are dishonest, and they wish to convince Bob that a message $b' \neq b$ is valid. The forgers have complete control of the classical channel used to send the message, but not the quantum channel for the distribution of public keys: Bob always receives a correct set of public keys. (However, we do not assume the cheaters behave honestly during the key verification stage.) Naturally, the forgers can always prevent any message from being received, or cause Bob to reject a valid message, but we do not consider this to be a success for the cheaters.

Our scheme is applicable to a variety of cryptographic problems. For instance, Alice may wish to sign a contract with Bob such that Bob can prove to Judge Charlie that the contract is valid. In this case, Bob should accept the contract whenever $s_B < c_1 M$, and Charlie should accept whenever $s_C < c_2 M$. This problem can also be solved by a classical protocol [11], with the assistance of an anonymous broadcast channel and recipients with distinct private keys. However, it is much simpler to distribute public keys, and we can adapt many classical methods for doing so to our protocol. The swap-tests in the key distribution stage provide an explicit check for key indistinguishability which is implicit in most classical schemes.

The security proof for the second scenario is straightforward. In the worst case, the forger Eve has access to all $2t$ copies of each public key. By Holevo's theorem, Eve can acquire at most $2tn$ bits of information about each bit string k_b^i. When Alice sends the signed message, Eve may attempt to substitute a different $b' \neq b$ and (possibly) different values of the $k_{b'}^i$ to go with it. However, since she lacks at least $L - 2tn$ bits of information about any public key which Alice hasn't revealed, she will only guess correctly on about $G = 2^{-(L-2tn)}(2M)$ keys. Furthermore, if she wishes to change a bit for which she did not correctly guess a key's identity, she has only probability δ^2 of successfully revealing the bit. Each recipient measures M keys, so when $b \neq b'$, each recipient will find (with high probability) that at least $(1-\delta^2)(M-G) - O(\sqrt{M})$ public keys fail. We will pick c_2 so that $(1-\delta^2)(M-G) > c_2 M$, which means each recipient either receives the correct message, or rejects the message with high probability.

For the security proof in the first scenario, where Alice is dishonest, we will simplify to the case where there are only two recipients, Bob and Charlie, but the proof can easily be generalized to $t > 2$ recipients.

Here, Alice wishes Bob to accept the message and Charlie to reject it or vice-versa. She can prepare any state she wishes for the public keys, including entangled states and states outside the family $|f_k\rangle$. For instance, she can prepare a symmetric state, such as $|\psi\rangle_B|\phi\rangle_C + |\phi\rangle_B|\psi\rangle_C$. Because this state is invariant under swaps, it always passes all tests, so that Bob and Charlie believe they have the same key. But that is an illusion — clever trickery by Alice who can nevertheless arrange that they disagree on the validity of the corresponding private key k_b^i. However, Alice cannot control which of them receives the valid key; it goes randomly to Bob or Charlie. Thus, since M is large, the difference $|s_B - s_C|$ is $O(\sqrt{M})$ with high probability, which makes it very unlikely that Bob and Charlie will get definitive but differing results. That is, when one of them (say, Bob) accepts a message, that is $s_B < c_1 M$, Charlie almost never rejects it, which would happen if $s_C > c_2 M$. The gap between $c_1 M$ and $c_2 M$ protects them against Alice's machinations.

Let us now prove this in general. Our goal is to compute the probability p_{cheat} that Alice can pass all the swap-tests but achieve $|s_B - s_C| > (c_2 - c_1)M$, meaning that Bob and Charlie disagree about the validity of the message. We do this by studying a global pure state $|\Psi\rangle$, which describes all of the public keys as well as any state that Alice may have which is entangled with the keys. Any state which passes the initial swap-tests will be symmetric between the test keys and the kept keys; in fact, it is symmetric between any individual test key and the corresponding kept key. Therefore, we can safely assume Alice prepares $|\Psi\rangle$ with this property.

Now, for each set of keys, the most general state is a superposition of two types of terms. A type-1 term passes the swap test, but leaves Bob and Charlie in agreement, on average, about the validity of the keys, while a type-2 term frequently fails the swap test. To perform the decomposition, we expand both the kept keys and the test keys in the basis $|f\rangle|f\rangle$, $|f_\perp^b\rangle|f_\perp^c\rangle$, $|+^a\rangle$, and $|-^a\rangle$, where the first ket is Bob's, the second is Charlie's, the states $|f_\perp^a\rangle$ form an orthonormal basis with $|f\rangle$, and $|\pm^a\rangle = |f\rangle|f_\perp^a\rangle \pm |f_\perp^a\rangle|f\rangle$. A type-1 term is any term for which both the kept and test keys are in a state $|f\rangle|f\rangle$, $|f_\perp^b\rangle|f_\perp^c\rangle$, or $|+^a\rangle$. Note that any sum of type-1 terms will always pass the swap test, but also has equal amplitudes for Bob and Charlie to pass key verification. A type-2 term is any term including a $|-^a\rangle$ state for the kept keys, the test keys, or both. For the type-2 terms, we explicitly note the symmetry between the kept and test keys, meaning the superposition $(|+^a\rangle|-^b\rangle + |-^a\rangle|+^b\rangle)/\sqrt{2}$ is the only way $|-^a\rangle|+^b\rangle$ can appear. That is, any sum of type-2 terms respecting this symmetry must have at least a 50% chance of failing the swap test. On the other hand, some superpositions of type-

2 terms can give different chances for Bob and Charlie to pass key verification.

Expanding every set of keys in $|\Psi\rangle$ in this way gives a global state which we can again divide up into two terms: $|\Psi_1\rangle + |\Psi_2\rangle$. Every summand in $|\Psi_1\rangle$ contains at most r type-2 tensor factors, where $r = (c_2 - c_1 - c)M$ for some constant $c > 0$; the rest are type-1 terms. Each type-1 term has equal amplitude to contribute to s_B and s_C, so the tensor product of $M-r$ such terms has a Gaussian distribution of amplitudes, centered at $s_B = s_C = (M-r)/2$ and with width $O(\sqrt{M})$. That is, most of the weight of $|\Psi_1\rangle$ falls on cases where $|s_B - s_C| \leq r + O(\sqrt{M}) < (c_2 - c_1)M$. $|\Psi_2\rangle$ consists of terms with more than r type-2 tensor factors. Since each type-2 term has at least a $1/2$ chance of failing the swap test, $|\Psi_2\rangle$ passes with probability no larger than 2^{-r}. Note that $|\Psi_1\rangle$ need not be orthogonal to $|\Psi_2\rangle$.

Now we can put this together to obtain a bound on p_{cheat}. The $|\Psi_1\rangle$ term might have a good chance of passing all swap tests, but yields an exponentially small chance of giving the required separation between s_B and s_C. The $|\Psi_2\rangle$ term might have $O(1)$ probability of having $|s_B - s_C| > (c_2 - c_1)M$, but only has an $O(2^{-r})$ chance of passing all swap tests. The best case for constructive interference between the two terms gives a total probability p_{cheat} at most twice the sum of the two probabilities for $|\Psi_1\rangle$ and $|\Psi_2\rangle$, which is still exponentially small in M. Therefore, Alice has $p_{\text{cheat}} \sim O(d^{-M})$ probability of successfully cheating for some $d > 1$.

Multi-bit messages can be sent by repeating the above process, using M pairs of public keys for each message bit. However, a much more efficient procedure is to first encode the message in a classical error-correcting code with distance M, and to use a single pair of public keys for each encoded bit. The previous protocol can be viewed as a special case of this using a repetition code. Valid messages are codewords of the error-correcting code; to change from one valid message to another requires altering M bits. Therefore, the above security proof holds with only two changes: G, the number of keys successfully guessed by Eve, is now $2^{-(L-2tn)}(2N)$, where N is the length of the full encoded message. In addition, if Alice attempts to cheat, she can produce a difference $|s_B - s_C| = O(\sqrt{N})$ with type-1 terms. We should thus have M scale linearly with N when the latter is very large.

Note that in a purely classical scheme, the public key can be given out indiscriminately. This cannot be true of a quantum scheme: when there are very many copies of a public key, sufficiently careful measurements can completely determine its state, and therefore one may as well treat the public key as classical. In that case, security must be dependent on computational or similar assumptions. Thus, any quantum digital signature scheme will necessarily require limited circulation of the public key.

The digital signature scheme provided here has many potential applications. It combines unconditional security with the flexibility of a public key system. An exchange of digital signature public keys is sufficient to provide authentication information for a quantum key distribution session. Quantum digital signatures can be used to sign contracts or other legal documents. In addition, digital signatures are useful components of other more complex cryptographic procedures.

One particularly interesting application is to create a kind of quantum public key cryptography. If Bob has Alice's public key, but Alice has nothing from Bob, then Bob can initiate a quantum key distribution session with Alice. Bob will be sure that he is really talking to Alice, even though Alice has no way to be sure that Bob is who he says he is. Therefore, the key generated this way can be safely used to send messages from Bob to Alice, but not vice-versa.

We have demonstrated the existence of an absolutely secure public key digital signature scheme, something which is not possible classically. Many potential improvements remain, however. A disadvantage of our protocol is that it consumes several key bits for each message bit. This makes key management unpleasant. Classical schemes allow reuse of keys and similar capability would be desirable in an improved quantum signature scheme. Other future goals, in addition to optimizing the protocol, would be to devise a method for efficiently distributing new public keys or signing certain quantum states (although it is not possible to sign a general quantum state). Also intriguing would be generalization of quantum one-way functions, to further exploit unique properties of quantum information for cryptographic purposes.

[1] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information* (Cambridge University Press, Cambridge, UK, 2000).

[2] R. Rivest, in *Handbook of Theoretical Computer Science* (Elsevier, Amsterdam, The Netherlands, 1990), Vol. 1, pp. 717–755.

[3] R. L. Rivest, A. Shamir, and L. Adleman, Comm. Assoc. Comput. Mach. **21**, 120 (1978).

[4] P. W. Shor, SIAM J. Comp. **26**, 1484 (1997).

[5] J. Rompel, Proc. 22th Ann. ACM Symp. on Theory of Computing (STOC '90) 387 (1990).

[6] L. Lamport, Technical Report CSL-98, SRI International (1979).

[7] H. Buhrman, R. Cleve, J. Watrous, and R. de Wolf, arXive e-print quant-ph/0102001 (2001).

[8] A. S. Holevo, Rep. Math. Phys. **12(2)**, 273 (1977).

[9] H.-K. Lo and H. F. Chau, Phys. Rev. Lett. **78**, 3410 (1997).

[10] D. Mayers, Phys. Rev. Lett. **78**, 3414 (1997).

[11] D. Chaum and S. Roijakkers, Lecture Notes in Computer Science **537**, 206 (1991).

Acknowledgements: DG was supported by a Clay long-term CMI prize fellowship. ILC was supported in part by the Things That Think consortium. We thank C. Bennett, D. DiVincenzo, D. Leung, H. K. Lo, M. Mosca, J. Smolin, B. Terhal, and W. van Dam for helpful comments.

Author electronic addresses: gottesma@eecs.berkeley.edu; ichuang@media.mit.edu